A Method for Prefetching Recursive Data Structure Traversals

Field of the Invention

This invention addresses the problem of prefetching indirect memory references commonly found in applications employing pointer-based data structures such as trees, linked lists, and graphs. More specifically, the invention relates to a method for pipelining traversals on these data structures in a way that makes it possible to employ data prefetching into high speed caches closer to the CPU from slow memory. It further specifies a means of scheduling prefetch operations on data so as to improve the throughput of the computer system by overlapping the prefetching of future memory references with the execution of previously cached data.

Background of the Invention

Modern microprocessors employ multiple levels of memory of varying speeds to reduce the latency of references to data stored in memory. Memories physically closer to the microprocessor typically operate at speeds much closer to that of the microprocessor, but are constrained in the amount of data they can store at any given point in time. Memories further from the processor tend to consist of large dynamic random access memory (DRAM) that can accommodate a large amount of data and instructions, but introduce an undesirable latency when the instructions or data cannot be found in the primary, secondary, or tertiary caches. Prior art has addressed this memory latency problem by prefetching data and/or instructions into the one or more of the cache memories through explicit or implicit prefetch operations. The prefetch operations do not stall the processor, but allow computation on other data to overlap with the transfer of the prefetch operand from other levels of the memory hierarchy. Prefetch operations require the compiler or the programmer to predict with some degree of accuracy which memory locations will be referenced in the future. For certain mathematical constructs such as arrays and matrices, these memory locations can be computed *a priori*. In contrast, the memory reference patterns of the traversals of certain data structures such as linked lists, trees, and graphs are generally unpredictable because the nodes that make up the graph are frequently allocated at run time.

In modern transaction processing systems, database servers, operating systems, and other commercial and engineering applications, information is frequently organized in trees, graphs, and linked lists. Lack of spatial locality results in a high probability that a miss will be incurred at each cache in the memory hierarchy. Each cache miss causes the processor to stall while the referenced value is fetched from lower levels of the memory hierarchy. Because this is likely to be the case for a significant fraction of the nodes traversed in the data structure, processor utilization will suffer.

The inability to compute the address of the next address to be referenced makes prefetching difficult in such applications. The invention allows compilers and/or programmers to restructure data

"O.K. TO ENTER" Clr. 8/4/05